CMR-1168 Orbital backtracking issue

The two possible interpretations of start circular latitude and orbits are given (Dan's rendering) in figure 1.

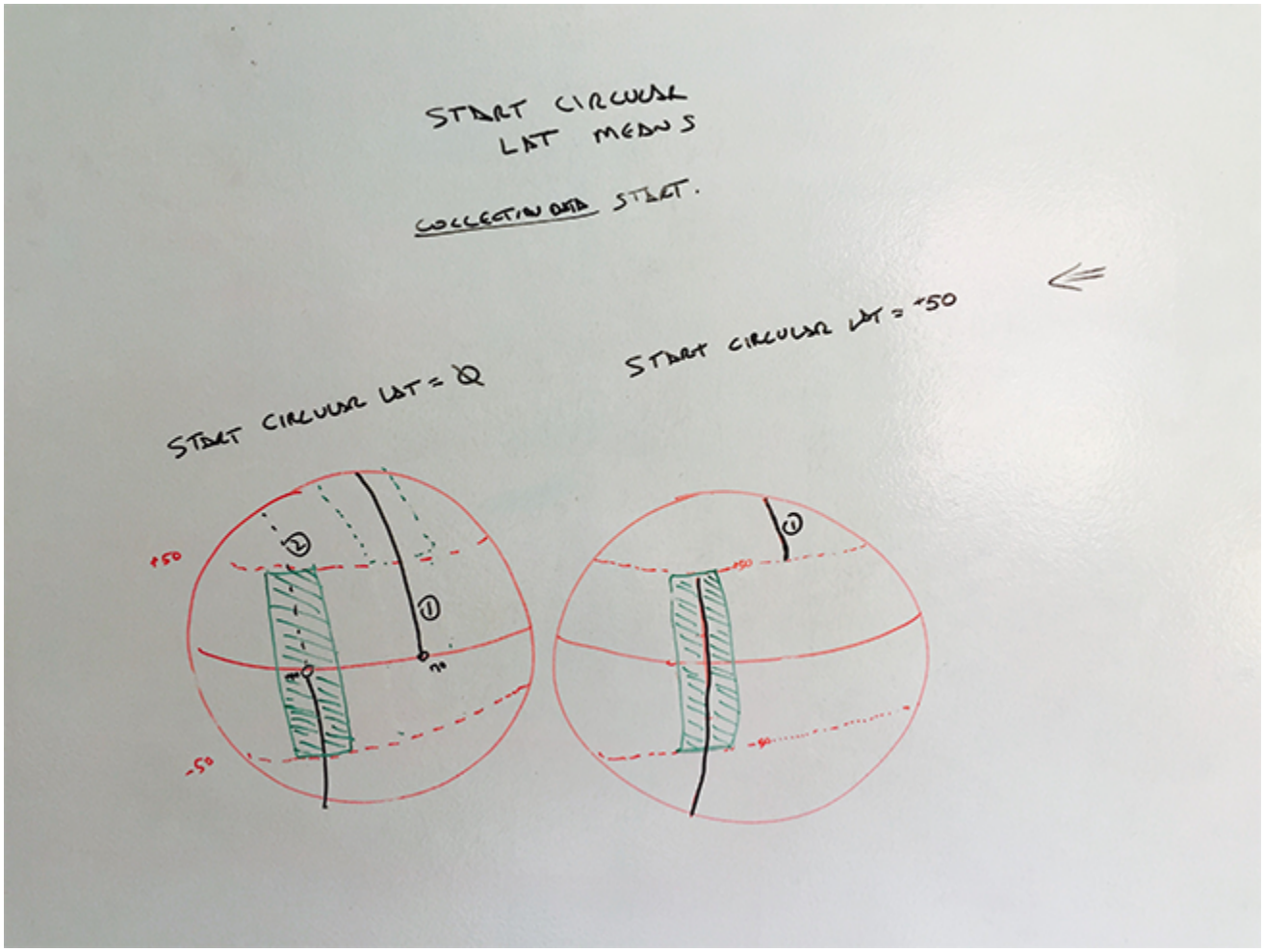

Figure 1 - Alternate interpretations of collection start circular latitude

On the left drawing, orbits begin at the equator and fractional orbit granules like the shaded region will lie on more than one orbit if they cross the equator. In the right drawing, orbits begin at the designated start circular latitude and a fractional orbit granule can never lie on more than one. The backtracking algorithm as implemented in the CMR/ECHO is based on the second interpretation.

Jason provided the following from 2008:

```
Ross swick info from 11003537


I found during testing of NCR 11003519 that I could not find a GLA05 granule after it was ingested.  It
turns out that the parameters set in the NSIDC XSLT were incorrect.   Below is an email from Ross Swick
with the changes to apply.


From Ross Swick:

We took a look at the granule you referenced and found this bit of

metadata in the .met file:
```

```
OBJECT = AdditionalAttributesContainer
            Data_Location = "NONE"
            Mandatory = "TRUE"
            Class = "2"
            OBJECT = AdditionalAttributeDatatype
                Data_Location = "MCF"
                Mandatory = "TRUE"
                Class = "2"
                NUM_VAL = 1
                Value = "int"
                TYPE = "STRING"
            END_OBJECT = AdditionalAttributeDatatype
            OBJECT = AdditionalAttributeDescription
                Data_Location = "MCF"
                Mandatory = "TRUE"
                Class = "2"
                NUM_VAL = 1
                Value = "Number assigned for the specific latitude
segment (1 = +50 to +50, 2 = +50 to -50, 3 = -50 to -50, 4 = -50 to +50)
of the track for the data."
                TYPE = "STRING"
            END_OBJECT = AdditionalAttributeDescription
            OBJECT = AdditionalAttributeName
                Data_Location = "MCF"
                Mandatory = "TRUE"
                Class = "2"
                NUM_VAL = 1
                Value = "Track_Segment"
                TYPE = "STRING"
            END_OBJECT = AdditionalAttributeName
            GROUP = PhysicalParameterDetails
                Class = "2"
                OBJECT = ParameterUnitsofMeasurement
                    Data_Location = "MCF"
                    Mandatory = "TRUE"
                    NUM_VAL = 1
                    Value = "counts"
                    TYPE = "STRING"
                END_OBJECT = ParameterUnitsofMeasurement
```

```
                OBJECT = ParameterRangeBegin
                    Data_Location = "MCF"
                    Mandatory = "TRUE"
                    NUM_VAL = 1
                    Value = "1"
                    TYPE = "STRING"
                END_OBJECT = ParameterRangeBegin
                OBJECT = ParameterRangeEnd
                    Data_Location = "MCF"
                    Mandatory = "TRUE"
                    NUM_VAL = 1
                    Value = "4"
                    TYPE = "STRING"
                END_OBJECT = ParameterRangeEnd
            END_GROUP = PhysicalParameterDetails
        END_OBJECT = AdditionalAttributesContainer
```

So it looks like "Track_Segment" is just an indicator of which quarter of the orbit this file is. They vary from 1-4 with segment 1 starting at 50 North (+50) ascending, going (nearly) over the North pole, ending at 50 North (+50) descending. Segment 2 starts at 50 North (+50) descending, crosses the equator descending, and ends at 50 South (-50) descending. Segment 3 starts at 50 South (-50) descending, goes (nearly) over the south pole, and ending at 50 South (-50) ascending. The file you are concerned about is a segment 4 file. It starts at 50 South (-50) ascending, crosses the equator ascending, WHICH IS THE ORBIT BOUNDARY, and ends at 50 North (+50) ascending.

SO your xslt is nearly correct:

```
> <xsl:variable name="trackNumber"
>>>                                                          select="../PSAs/PSA[PSAName =
'Track_Segment']/PSAValue" />
>>>
>>>                                                  <xsl:choose>
>>>                                                          <xsl:when test="$trackNumber='1'">
>>>                                                                  <StartLat>50</StartLat>
>>>                                                                  <StartDirection>A</StartDirection>
>>>                                                                  <EndLat>50</EndLat>
>>>                                                                  <EndDirection>D</EndDirection>
>>>                                                          </xsl:when>
```

```
>>>                                                     <xsl:when test="$trackNumber='2'">
>>>                                                             <StartLat>50</StartLat>
>>>                                                             <StartDirection>D</StartDirection>
>>>                                                             <EndLat>50</EndLat>
>>>                                                             <EndDirection>D</EndDirection>
>>>                                                     </xsl:when>
>>>                                                     <xsl:when test="$trackNumber='3'">
>>>                                                             <StartLat>50</StartLat>
>>>                                                             <StartDirection>D</StartDirection>
>>>                                                             <EndLat>50</EndLat>
>>>                                                             <EndDirection>A</EndDirection>
>>>                                                     </xsl:when>
>>>                                                     <xsl:when test="$trackNumber='4'">
>>>                                                             <StartLat>50</StartLat>
>>>                                                             <StartDirection>A</StartDirection>
>>>                                                             <EndLat>50</EndLat>
>>>                                                             <EndDirection>A</EndDirection>
>>>                                                     </xsl:when>
>>>                                             </xsl:choose>
```

I wouldn't use trackNumber as the variable name if what you are keying off is the track SEGMENT. "Track Number" has another meaning so using it for the variable name here can lead to confusion.

Otherwise you just have a few sign errors. Everything starts and ends at plus OR minus 50 - you only have +50. Track segment 1 is correct. Segment 2 ends at -50, segment 3 starts and ends at -50, and segment 4 starts at -50.

The directions look correct.

So then using the algorithm you've got start and end circular latitudes like so:

Segment 1: (50, 130)
Segment 2: (130, 230)
Segment 3: (230, 310)
Segment 4: (310, 50)

Segment 4 is a bit problematic because it crosses the orbit boundary,

which is why there's the special case in the algorithm that Lei pointed

out. SO you actually end up with:

Segment 4: (310, 410)

And that is with the data indexed to the first crossing.

I checked and the indexed start and stop clats for the granule are indeed 310, 410. Initially I thought that we were in agreement that the second interpretation (rightmost drawing) is the correct one and that the problem is that the start clat for the collection is incorrectly set to -50 when it should be +50. Talking with Jason I am no longer convinced of this.

Next steps:

1. Change the start circular latitude for the collection to +50 and verify that this results in correct granules being found.
2. Come to an understanding of exactly why this was a problem and how this fixes it.
3. Determine if this is an isolated problem or if this impacts other collections.
4. Depending on the answer to 2, contact NSIDC to work out a plan to correct the metadata.
5. Add an integration test to compare Orbital Backtracking results to granule polygon search results.

For the granules the test should calculate polygons which will be matched against a set points covering the earth. Then searches

The integration test should do the following:

1. ingest the collections from production that contain orbit information as well as their granules.
2. Compute polygons for all the granules.
3. Iterate over a set of points that cover the earth and for each point
    a. Find the granules whose polygons intersect the point
    b. Do a spatial search to get all the returned granules for the point (these are the backtracking results)
    c. Compare the two sets and fail if they are different

Update: Leo and James tried setting the start circular latitude for the collection to +50 deg and the false positive still occurred. Using 0 deg did not work either.

It is unclear how the collection StartCircularLatitude is used in the orbit back tracking algorithm. I can see how the back tracking algorithm works for whole multiple orbits (where the echo-orbits denormalizeLatitudeRange function finds ranges of circular latitude for the orbit that would satisfy the latitude condition of the search area, and the granule equator crossing longitude matching against the equator crossing longitude condition calculated from the back tracking algorithm), but not sure how it could possibly work for partial multiple orbits.

Update 2: Patrick has determined that the issue lies with the search area crossing the start circular latitude for the collection. The search area needs to be split into two areas, one above the start circular latitude and one below it. The top area will have one crossing range plus (start circular latitude, top latitude) as its latitude range and the bottom area will a different crossing range plus (bottom latitude, start circular latitude) as its latitude range. This will require changes to the echo_orbits library and the CMR Clojure code that generates the orbital back tracking search condition.

Also, Patrick has determined the problem with the KML generation for granules is due to a simple sign error. This fix is being incorporated into the changes to fix the start circular latitude bug.